*Int. J. Appl. Math. Comput. Sci., 2011, Vol. 21, No. 2, 259–274*
DOI: 10.2478/v10006-011-0019-9

# RESILIENT CRITICAL INFRASTRUCTURE MANAGEMENT WITH A SERVICE ORIENTED ARCHITECTURE: A TEST CASE USING AIRPORT COLLABORATIVE DECISION MAKING

MARTIN HALL-MAY *, MIKE SURRIDGE *, ROMAN NOSSAL-TÜYENI **

* IT Innovation Centre
Gamma House, Enterprise Road, Southampton S016 7NS, UK
e-mail: {mhm,ms}@it-innovation.soton.ac.uk

** Austro Control
Österreichische Gesellschaft für Zivilluftfahrt mbH
Schnirchgasse 11, A-1030 Vienna, Austria
e-mail: roman.nossal-tueyeni@austrocontrol.at

The SERSCIS approach aims to support the use of interconnected systems of services in Critical Infrastructure (CI) applications. The problem of system interconnectedness is aptly demonstrated by 'Airport Collaborative Decision Making' (A-CDM). Failure or underperformance of any of the interlinked ICT systems may compromise the ability of airports to plan their use of resources to sustain high levels of air traffic, or to provide accurate aircraft movement forecasts to the wider European air traffic management systems. The proposed solution is to introduce further SERSCIS ICT components to manage dependability and interdependency. These use semantic models of the critical infrastructure, including its ICT services, to identify faults and potential risks and to increase human awareness of them. Semantics allow information and services to be described in a way that makes them understandable to computers. Thus when a failure (or a threat of it) is detected, SERSCIS components can take action to manage the consequences, including changing the interdependency relationships between services. In some cases, the components will be able to take action autonomously, e.g., to manage 'local' issues such as the allocation of CPU time to maintain service performance, or the selection of services where there are redundant sources available. In other cases the components will alert human operators so they can take action instead. The goal of this paper is to describe a Service Oriented Architecture (SOA) that can be used to address the management of ICT components and interdependencies in critical infrastructure systems.

**Keywords:** resilience, QoS, SOA, critical infrastructure, SLA.

## 1. Introduction

The SERSCIS ('Semantically Enhanced, Resilient and Secure Critical Infrastructure Services') approach aims to support the use of interconnected ICT systems used to plan and manage operations in critical infrastructures such as airports. Failure or underperformance of any of the interlinked ICT systems owing to faults, mismanagement or (cyber-)attack compromise the ability of any or all the interconnected businesses to plan their use of resources, to maintain high levels of efficiency, and to continue providing information needed by others. The SERSCIS approach is to develop service-oriented technologies for creating, monitoring and managing ICT systems, allowing dynamic adaptation to manage changing situations, and to counter the risk amplification effect of interconnectedness. This can be evaluated by investigating failure scenarios caused by or impacting ICT systems, to show how SERSCIS provides more accurate risk assessments and thereby allows the impact of such failures to be minimised.

The technical approach used by SERSCIS is to treat each ICT component as a service, which has a specification of its dependability properties incorporated into a Service Level Agreement (SLA) with its users (humans or other services). This allows interconnections between services to be managed to maintain the dependability of the overall system in which they are used. The overall system is then able to specify its own dependability characteristics for the services it in turn provides to its users.

The key objectives of the SERSCIS approach are the following:

- to devise ways to encode dependability commitments in a machine readable way, so they can be included in SLAs;
- to develop service governance mechanisms to ensure these commitments will be met, using autonomous monitoring and management of available resources (which may themselves be services), and adaptive workflow technology to orchestrate and utilise these resources;
- to verify the approach and its impact on best practice in an application scenario in air transportation.

A certain amount of 'machine intelligence' is needed to understand SLA commitments and to govern and orchestrate services according to this approach. The approach therefore makes extensive use of semantic modelling and reasoning methods to underpin machine understandable dependability specifications and decision making. These models are also used to analyse ICT dependencies and threats and to provide decision support for human end-users during system design, deployment and operation.

## 2. Related work

At IT Innovation, the GRIA middleware (*GRIA.org*, 2011) forms a central vehicle for our research and understanding of a service oriented infrastructure. Originally developed with the FP5 RTD Project on *Grid Resources for Industrial Applications*, GRIA was the first grid middleware designed to support computational service provision between independent organisations that need not be part of a virtual organisation or other community. Since its first release in 2004, GRIA has become a vehicle for conducting further research and disseminating (as open source software) new results in a multi-stakeholder service oriented infrastructure. Insights gained from GRIA provide the fundamental basis for the SERSCIS approach to developing strategies and mechanisms to model and manage risk and interdependency in increasingly autonomic services and infrastructures. SERSCIS is also related to other previous work by the authors, namely, innovations in dynamic security, workflow and service management using the bilateral SLA that were developed as part of the NextGRID project (*NextGRID*, 2011).

In its aims, the SERSCIS approach also builds on the successes of a number of other projects. Research within the DIRC project and its successor InDeED (Sommerville *et al*., 2007) raised the need to tackle dependability throughout the lifecycle of a system, including its interactions with humans. CRUTIAL (Verissimo *et al*., 2008) used a Petri net and state modelling to understand the operation, behaviour and failure modes of the CI in systems of systems arrangements. The IRRIIS project (Balducelli *et al*., 2008) is also improving CI dependability through middleware improved technology and simulation in synthetic environments. There has been work on fault and sabotage tolerance using an SLA for grids (Hovestadt, 2005; Naqvi *et al*., 2008), which SERSCIS builds on to create resilient service-oriented infrastructures.

DeDiSys (Froihofer *et al*., 2007) aims to manage the trade-off between availability and meeting service constraints in loosely coupled grid or P2P systems. SERSCIS aims to manage a similar dependability trade-off through a framework of automated service governance, including SLA negotiation, dynamic resourcing as well as runtime Quality of Service (QoS) and Quality of Experience (QoE) monitoring, along with adaptive workflow composition, decision support and risk-aware system modelling techniques. SERSCIS is relevant to current research on quantitative risk assessment, such as that by Mikulik and Zajdel (2009), as it provides a means to monitor service-oriented infrastructures for the occurrence of threats and effectiveness of mitigation strategies identified as part of the safety assessment process.

## 3. SERSCIS components

We propose a service-oriented architecture with SLA-based management that builds on semantic models. This architecture comprises the following four main areas of component technology.

**3.1. System modelling.** System modelling covers the development of (semantic) models of critical infrastructure requirements and behaviour, including the ICT components. These models are used throughout the SERSCIS framework as computer-understandable descriptions that provide the basis for automated, dependable ICT operation and feedback. The Web Ontology Language, OWL-DL, is used to model ontologies describing critical infrastructure aspects. Models contain valuable system knowledge as well as performance and pricing models (Benkner and Engelbrecht, 2006).

**3.2. System governance.** System governance covers the development of monitoring mechanisms and management actions and policies to maintain the dependability (including security and trust relationships) exhibited by SERSCIS-enabled services. The service-oriented infrastructure middleware GRIA (Surridge *et al*., 2005) is used as a basis for the governance technological framework. The emphasis is on the controlling available resources such that dependability requirements can be met, and where the system provides services, describing their

non-functional characteristics in (semantically tractable) Service Dependability Agreements (SDAs).

**3.3. System composition.** System composition covers the development of automated composition and orchestration of services to implement workflows at the system level to meet dependability requirements, using SDA terms to control the selection of services from those made available by the system governance mechanisms. The emphasis is on development of dynamically adaptive workflow orchestration mechanisms based on semantic descriptions of workflows, dependability requirements, and (via the SDA) available resources.

**3.4. Decision support.** Decision support covers the provision of tools to present information to human operators of a critical infrastructure and its associated ICT. SERSCIS will make use of autonomic service management models driven by high-level policies supplied by the operators, who may also be involved in initiating or carrying out management actions. It is therefore necessary to provide tools to help ICT implementers understand how a SERSCIS-enabled network will behave. Decision support tools aid operators that decide how to deploy applications in defining high-level policies in a SERSCIS-enabled framework, and in understanding the resulting (dynamically adaptive) behaviour that changing these policies has in an operating critical infrastructure.

## 4. Critical infrastructure management

The key to SERSCIS is that it helps to manage risks and interdependency in the use of ICT systems within a critical infrastructure, by adapting the ICT composition in response to events. Management in this context is concerned with sending controlling signals to the critical infrastructure ICT components when monitoring data indicate a need to do so.

Without SERSCIS, the ICT operators can of course monitor information supplied by ICT components used within the critical infrastructure, and take action when they consider this information indicates a need to do so. This provides a 'humanised' or 'slow' management loop between the operators and the infrastructure, indicated in the upper half of Fig. 1. However, without SERSCIS, the interconnectedness of ICT systems used makes human decision making very difficult, as problems (or actions taken) elsewhere may affect the quality of information available with which to make decisions. Moreover, any action a human operator takes may have an adverse impact elsewhere. Thus the risk of incorrect responses and the damage this might do are both increased. One of the key goals of SERSCIS is to address this 'risk amplification' effect from ICT interconnectedness.

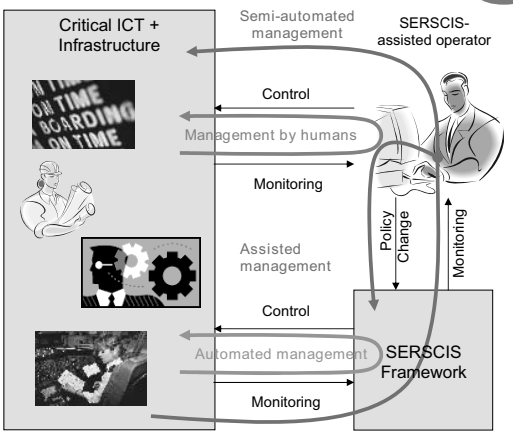


Fig. 1. SERSCIS interactions with a critical infrastructure and operators.

The SERSCIS framework monitors the critical infrastructure and uses a common Web service management interface to manage the dynamic composition of services and underlying resources. This process represents an automated management loop (marked as 'automated management' in Fig. 1), in which the SERSCIS framework takes action as and when required by the management policy. The latter is defined by a SERSCIS-assisted operator and may be dynamically updated.

In addition to the above, SERSCIS governance components may conclude that some action is required that cannot be implemented autonomously. In such cases, a signal is sent to the human operators. These signals may simply advise the operator that a management action may be needed, leaving the human to decide what action to take (if any). In some cases, the signal may also propose the action, but leave the human to decide whether or how to carry it out. This provides an 'assisted' management loop, which is also shown in Fig. 1.

The operators can also provide control inputs to the SERSCIS framework, e.g., to change the models it uses to analyse the critical infrastructure, or to change the range of monitoring inputs or automatic actions available to it. These control inputs do not directly affect the critical infrastructure itself, but they do change the way SERSCIS uses agile service composition models to support its future management. It is important to recognise that this facility to define SERSCIS models and policies is relevant even before the critical infrastructure and associated ICT are deployed, as well as during its operation. In the predeployment phase, this interaction can be used to support the ICT implementers, helping them to design and configure interconnected ICT systems in a way that allows risks to be managed by design as well as through subsequent adaptation.

## 5. High-level service architecture

The management of critical infrastructure ICT components and interconnections is considered an integral part of the overall SERSCIS approach. This is addressed by treating all ICT components as services, whose dependability can be specified via machine-understandable SDAs, allowing automatic and semi-automatic management of ICT dependability and interdependence, along with the rest of the critical infrastructure. Thus the SERSCIS framework from Fig. 1 contains the following:

- services whose purpose is to establish and keep track of the SLAs that specify how critical infrastructure ICT services should interact;
- services to monitor each critical infrastructure ICT component and to ensure (through automated or assisted control actions) that it behaves in accordance with its SLA;
- services or other components to support dynamic adaptation of critical infrastructure ICT, especially its access control policies, interconnectedness and resourcing levels.

The high-level architecture provides a preliminary decomposition of these SERSCIS framework facilities into software components, and explains how they interact with the critical infrastructure services they monitor and manage. To derive this architecture, we first consider how the lifecycle of SLAs (which are new entities introduced by SERSCIS) should relate to the critical infrastructure ICT services (the application to which SERSCIS is being applied).

This will lead to a decomposition of the SERSCIS framework needed by each service provider into components for managing the SLA, the critical infrastructure services, the resources available to those services, and the way services are composed and interact with each other.

## 6. SLA lifecycle

The most important architectural issue in developing for resilience is to have a common understanding regarding the lifecycle of SLAs, services and resources, a lesson learned previously in the GRIA project (Surridge *et al*., 2005). Unfortunately, these lifecycles are only loosely coupled, which leads to a large variety of possible scenarios as shown in Fig. 2. SERSCIS builds on the SLA definition and lifecycle developed under the NextGRID project (Hasselmeyer *et al*., 2007) as implemented in the GRIA middleware (Boniface *et al.*, 2006; 2009).

In Fig. 2, the following four main state models are illustrated:

- the service itself: its definition including its implementation as a piece of software and description via models of its behaviour, configuration (by specifying management models) for use at a service provider, and deployment to make it executable using allocated resources;
- resources: their acquisition by the service provider, and their allocation (or deallocation) for use by a particular service according to the provider's management policies;
- the SLA offer (or template): the specification by the service provider of terms (including dependability commitments) under which they can make the service accessible, and the publication of these terms to potential consumers;
- the interaction between a service provider and a service consumer: this includes the initial request for an SLA based on a published template/offer, the granting (or otherwise) of the request leading to an SLA being made between the two, and enabling access to the service under this SLA.

The solid lines in Fig. 2 represent state transitions in each of these processes. The loose coupling between the processes is indicated by the dashed arrows linking different processes, which signify that one process must be in a given state for the other process to enter a given state. For example, consumer interactions have a state where the service is accessible to the consumer. This state can only be reached if the service is deployed at the service provider, which is denoted by the dashed arrow from the 'Service Deployed' state in the service lifecycle to the 'Service Accessible' state in the consumer interaction lifecycle. To summarise these couplings,

- the service must be defined before an SLA template (specification of offered dependability commitments) can be formulated;
- the SLA template must be published before potential consumers can request an SLA embodying its commitments;
- the service must be configured with a management policy before resources can be allocated to it;
- at least some resources must be allocated before a service can be deployed;
- a service must be deployed before it can be made accessible to consumers.

The main purpose of the SERSCIS framework is to support the use of the SLA in consumer interactions with a service, including the introduction of service offers and SLA creation, and to facilitate coupling between consumer interactions, service management and resource

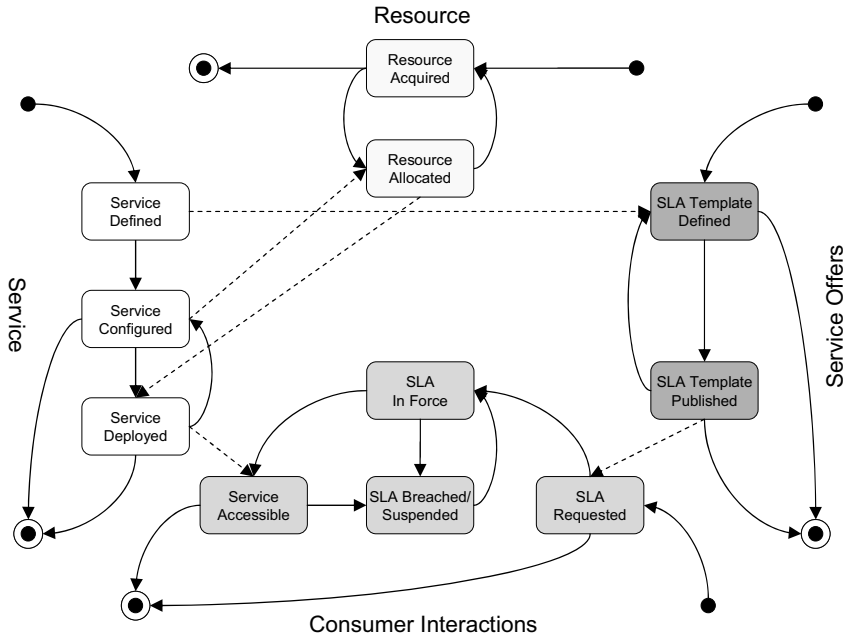


Fig. 2. Lifecycle models.

management, keeping them all consistent with the dependability commitments made in the SLA (Hovestadt, 2006; Hasselmeyer *et al.*, 2006).

The loose couplings represented by dashed arrows in Fig. 2 allow for many possible scenarios. For example, at one extreme the provider may procure and allocate resources and deploy the service long before they start to define SLA templates to specify their dependability commitments. This may be a common scenario where a service is already operating and the provider decides to introduce SERSCIS technology to manage its dependability.

At the other extreme, the provider may acquire the service implementation (i.e., the software), then create and publish SLA templates, and wait until they have agreed an SLA with a consumer before even configuring the service. This scenario must be followed where service providers are willing to negotiate over the exact service levels and other specifications with consumers. In such cases, the SLA template includes service metrics but not agreed levels, and the policies to regulate services have to be configured after the SLA is negotiated. In practice, it is difficult and/or costly for a service provider to manage resources and services against a large number of potentially different commitments (McKee *et al.*, 2007). In SERSCIS, we currently consider only the case where the management policy is defined first, and a limited (discrete) set of SLA templates are derived from it, which encode non-negotiable dependability commitments that the management policy is able to meet.

Finally, although Fig. 2 does not explicitly show cardinality, in practice the couplings can be many-to-many. For example, one SLA may cover several services and one service may be accessible to different consumers under many different SLAs. Similarly, services may need multiple resources, and multiple services may share resources, and so on. Keeping track of these relationships is one of the key requirements for a dynamic, (semi-)autonomous management architecture.

## 7. Service provider architecture

Given the above model for the lifecycle, it is possible to define a high-level service provider architecture for SERSCIS framework components to manage services (and consumer interactions), the SLA and resources during each phase of their respective lifecycles, as shown in Fig. 3.

The software components include the service interface and workflow orchestrator, along with SERSCIS governance components to support the management of services, resources and the SLA. These components include

- a security service access control point, able to restrict access to the service according to a security policy that must be dynamically updatable (Boniface *et al.*, 2005);
- an SLA manager that hosts SLA templates and handles requests from clients for the SLA based on them: the SLA manager grants new SLAs, provides information to the clients on their status, and may terminate existing SLAs if required;
- a service manager that monitors the quality of service

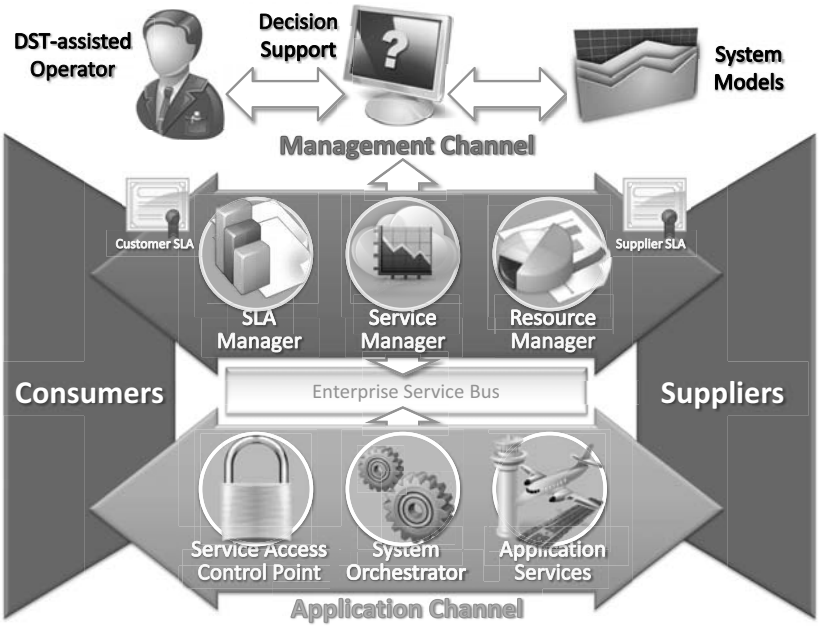


Fig. 3. High-level service-provider architecture.

and quality of experience reports and analyses them using the service model against the provider's service commitments and management policy, and initiates appropriate management actions;

- a resource manager that handles the acquisition/allocation and removal of resources, and maintains a registry (Radetzki *et al*., 2007) of these resources in which the orchestrator can discover resources when it has to execute a service workflow.

Governance components are deployed at each service provider that wishes to manage some or all of their services. In this way, governance remains distributed and light weight, without the need for a monolithic governance 'agent'. Components within each service provider communicate via an Enterprise Service Bus (ESB), which allows them to be loosely coupled. The relationship between these information sources and the lifecycle models from Fig. 2 are as follows:

- the system model and management policy are created by software developers and system administrators using SERSCIS ontologies, supported by SERSCIS modelling and decision support tools;

- service commitments are created or removed when an SLA is negotiated or terminated, based on an SLA template;

- the service access policy is dynamically generated when the service is deployed, and updated when an SLA is negotiated or terminated;

- the QoS event stream is generated by a service interface (or wrapper to a black-box service (Michlmayr *et al*., 2008)) based on what is provided or can be measured by the service;

- the QoE event stream is generated by the orchestrator when using other services based on what is measurable by or meaningful to the consumer of that service. Note that while QoS is a measure of what the service provides to its consumers, QoE is a measure of how its resources (including other services) have performed;

- resource registry entries are created or removed when resources are procured and allocated or de-allocated.

Actions taken by the service manager may induce changes in the available resources, according to the management policy of the provider. For example, it may ask the resource manager to negotiate additional SLAs to provide a greater resource volume or redundancy, or allocate more in-house resources. Alternatively, it may seek to manage demand on the service. For example, it may simply revoke the SLA template so that no new SLAs can be granted by the SLA manager, preventing any further increase in the level of service commitments. It may go further, by updating the security access policy to restrict access if the SLA allows this. It may even ask the SLA manager to breach or to terminate the SLA. The service itself may also support some management actions to influence its behaviour, and the service manager can use these if the management policy allows it. These actions can in

principle be taken automatically, implementing the 'automated' management loop from Fig. 1.

SERSCIS components also provide administration interfaces giving operators direct access to the service manager, SLA manager and resource manager, and to the associated models, policies and monitoring data. In many situations, the management policy will instruct the service manager to inform an operator that action is needed, leaving a human to decide whether and if so how to act. The operator can then implement their action directly on the critical infrastructure, by accessing SERSCIS components or by changing the models and policies used to control them. This provides the 'assisted' management loop in Fig. 1.

Finally, SERSCIS provides decision support facilities to help operators understand the behaviour of the system, based on events generated by the components shown in Fig. 3. As noted above, this facility will also be useful prior to the operation of the SERSCIS-enabled ICT infrastructure, allowing an analysis of risks using the service model during the design phase, and providing insights into the commitments that can be made and the management policies needed to meet them. These tools will also play a role in auditing processes used to analyse and verify/improve the management of the infrastructure.

A number of application services are managed by a single governance component, comprising a service manager, SLA manager and resource manager. The orchestration component coordinates workflows involving 'local' resources as well as resources encapsulated in services from other organisations. A SERSCIS-assisted operator has an overview of all the services within a given organisational/operational domain. The operator's situational awareness is supplemented by a decision support component, which uses a model of the whole system. The 'whole system' may include details of concepts outside of the operator's immediate control, i.e., involving the repercussions that local actions will have on the wider system of systems. These system models may be shared across organisational boundaries.

## 8. Layered approach

Figure 4 shows three concentric dependability management loops that arise from the SERSCIS architecture. Similar to Kramer and Magee's model for autonomic systems (Kramer and Magee, 2009), this allows runtime information to propagate up to system decision makers and governance actions to flow down to control service execution. System modelling and decision support components are closely linked and may run synchronously; however, with respect to governance and workflow components, the execution is entirely asynchronous.

The inner runtime loop runs at the speed of the application services, selecting the most appropriate resources

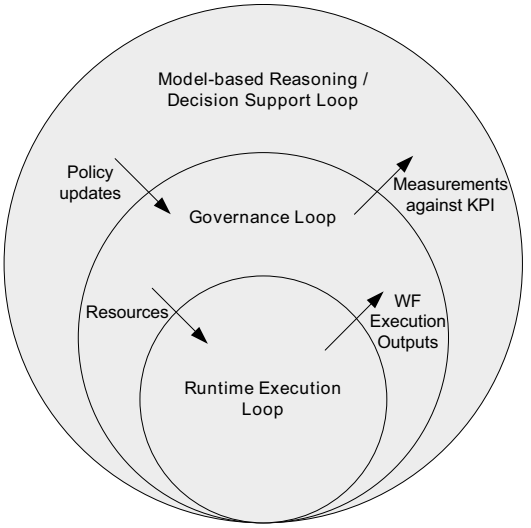


Fig. 4. Concentric dependability management feedback loops.

and executing the workflow. The governance loop monitors and manages resources against SLA commitments, at a slower rate. This loop has to be asynchronous with respect to application responses because it is often infeasible for the orchestrator to wait while a new SLA is negotiated and approved. Finally, the outer loop runs even more slowly so SERSCIS-assisted human operators can be involved, making changes to autonomic governance policies in response to changes in Key Performance Indicators (KPIs).

It is this separation of decisions, which simplifies the work that each set of components must do, that is the guiding principle of SERSCIS. Such a layered approach to the architecture was taken in recognition of real-world business practices and performance.

## 9. A-CDM test case

Airport Collaborative Decision Making (A-CDM) brings together the main airport partners—Air Traffic Control (ATC), airport operators, airlines, ground handlers, and Europe's Control Flow Management Unit (CFMU)—to share operational data. Such information sharing is fundamental to achieving a common situational awareness, which improves decision making (Aguado, 2009). The SERSCIS test case focuses on the process of 'turning round' an aircraft from the moment it arrives 'in block' to when it taxis out for take-off.

Airport collaborative decision making is an approach to optimizing resource usage and improving timeliness at an airport. It is about all partners at an airport working together, openly sharing accurate information and—based on the information—making decisions together. Through the use of A-CDM, the predictability of airport operations is improved. All actions involved in turning around an

aircraft can be planned more accurately, and the plans can more easily be controlled with respect to the actual operation. This yields a number of benefits:

- Usage of resources can be optimized. If every actor knows more accurately where and when human and physical resources are needed, resource scheduling is more easily able to meet these demands and to avoid bottlenecks while at the same time preventing extensive periods of idling resources.
- While it is not a goal to increase airport capacity by A-CDM, the concept allows better use of the available capacity. Through accurate planning, the waste of departure and arrival slots, a scarce and therefore precious resource at most major airports, is reduced.
- Accurate planning and continuous monitoring deviations enables countermeasures to be set proactively in case problems arise. Thus the timeliness of operations and—most importantly—timeliness of flights increase. This is beneficial for airlines and their customers alike.

All of these benefits are relevant at the local and regional level. Yet, A-CDM also has a European, network-wide perspective. The central flow management unit of EUROCONTROL monitors the capacity of airspace sectors and imposes restrictions by issuing the so-called 'slots' in case congestion might arise. Currently, this planning is mainly based on flight plan information that is filed up to three hours before the actual flight. Changes, in particular last minute changes, e.g., due to late passengers, are not taken into account. Hence, everyday a huge amount of airspace capacity is wasted due to inaccurate information. The airports applying A-CDM can more accurately determine the take-off time of departing flights

**9.1. Why is this scenario 'critical'?.** The focus of the scenario is on airside ground-handling activities during aircraft turnaround, exploiting collaborative decision making between the organisations and actors involved within an airport. This process is critical to the smooth and safe operation of European air transport because

- it determines the readiness of an aircraft for take-off, which has to be predicted reliably as an input to European air traffic management;
- it has to be adapted to cope with changes elsewhere in the air traffic infrastructure, e.g., delays to incoming flights, or disruption of ground handling resources such as supplies or staff;
- it provides opportunities for malicious attacks, such as Denial of Service (DoS), to damage European air transport.

Malicious attacks may be directed against ground handling resources, aiming to disrupt aircraft turnaround or to gain unauthorised access to the aircraft. They may also be directed against the ICT systems used for collaborative decision making, to disrupt ground handling operations or the flow of information between these operations and European air traffic management.

**9.2. Aircraft turnaround.** The turnaround of a single aircraft involves multiple actors with different roles:

- the airline, who operates the aircraft, schedules the inbound and outbound flights and ultimately pays for the turnaround services used;
- the European and local (airport) air traffic management services, who allocate and manage the use of airspace including airport landing and take-off slots, provide information about inbound flights, and need predictions when an aircraft will be ready for outbound flights;
- an A-CDM information service provider, which operates an A-CDM Information Sharing Platform (ACISP) for exchanging information between the airline, air traffic and ground handling service providers at the airport;
- the ground handling organisation, which coordinates the provision of services to the aircraft to make it ready for takeoff;
- the provider(s) of ground handling services: in our scenario these include aircraft cleaning, refuelling, baggage unloading and loading, and catering.

In the test case scenario, we consider that some (but not all) of the above actors provide their services in competition with each other. Some redundancy is therefore available in the airport and the premise of SERSCIS is that, by using an agile SOA, one can adapt the information sharing networks sufficiently rapidly to exploit this redundancy to improve overall dependability.

The idea here is that, without SERSCIS, the airline has to choose a ground handler and order turnaround service for each scheduled aircraft movement through the airport. The airline and air traffic service providers between them determine if and when the aircraft will land, and share this information with the ground handler via the ACISP. This enables the ground handler to manage its resources (including services from the four types of actors on the right of Fig. 5), to plan and execute the turnaround. Note that the airline must have a relationship with its ground handler, but it might not trigger ground handling actions directly except through the ACISP, so this relationship is therefore shown as a dashed line in Fig. 5. Airlines may also have their own fuelling, baggage, cleaning and

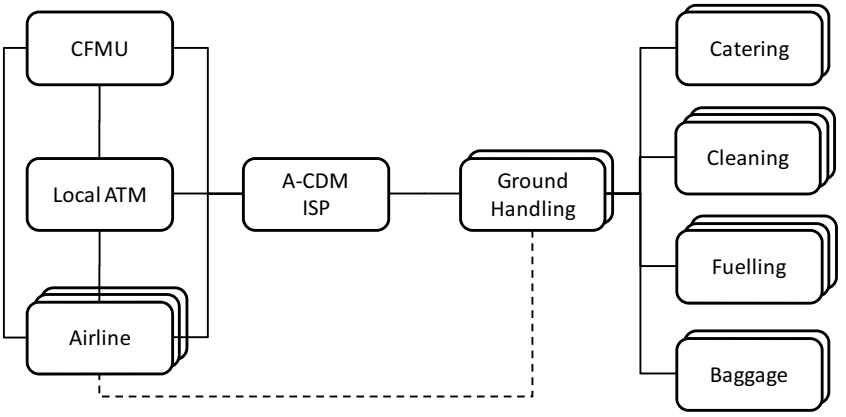


Fig. 5. Actors and services in the A-CDM test case.

catering services at some airports, but these relationships do not exist at all airports and for simplicity are not shown at all.

We assume here that there is only one A-CDM provider that allows information to be shared between the local and European ATM providers and multiple airlines and ground handling providers. The ground handler orchestrates the provision of the four air-side turnaround services each of which can also have multiple providers, as shown.

**9.3. Milestone approach.** In order to structure the turnaround process at an airport, EUROCONTROL has devised the so-called milestone approach. Each milestone defines a significant event along a timeline that shows the progress of a flight. The milestones cover the entire rotation, i.e., the inbound as well as the turnaround and the beginning of the outbound phase. Figure 6, taken from Aguado (2009), shows the milestones.

For more details on the milestones, please refer to the Airport CDM Implementation Manual (Aguado, 2009). Two milestones are of specific importance to the turnaround of an aircraft. Firstly, the estimated landing time (ELDT) and its actual value ALDT determine the beginning of the turnaround phase of a rotation. Adding the taxi time estimate to ELDT results in the estimated in-block time (EIBT). This time estimate is the main scheduling event for the ground handlers and their resources. As soon as an aircraft is in-block, ground handling must commence. Thus the ground handler has to ensure availability of personnel and equipment at the aircraft stand.

The second major milestone is the target off-block time (TOBT). This is newly introduced with A-CDM. This estimate or target is provided by the ground handler and states when he expects the aircraft to be ready for going off-block. It marks the limit between the turnaround and the outbound phase of a flight. Thus it forms the basis for all further calculations. From TOBT, the target take-off time (TTOT) is calculated and sent to the CFMU of EUROCONTROL in Brussels. As mentioned above, the CFMU optimises the network performance for the whole of Europe—a task that is substantially easier and provides better results given more accurate TOBT rather than the currently used *estimated* off-block time (EOBT) taken from ATC flight plans.

**9.4. SERSCIS test bed.** The focus of the test bed is on information flows complementing the turnaround for a specific subset of the full process. The test bed considers the information exchange between the CFMU, ATC, the ACISP, a ground handler and ramp service providers such as a fuelling company. ICT systems of the above mentioned actors are interconnected and exchange information vital to A-CDM.

At the same time, the interconnection between the ground handler and the ramp service providers has an underlying physical workflow that takes place at the airport apron. This workflow is represented by the information flow. The ground handler needs to inform the ramp service providers of the incoming aircraft well in advance so they can schedule their resources. When the aircraft is in the vicinity of the airport, the information by the ground handler is used to actually dispatch the resources of the respective ramp service provider. Upon completion of his job, a ramp service provider informs the ground handler, who can then trigger the next step in the turnaround process.

Figure 7 illustrates the turnaround of a long-haul aircraft. The example is adapted from turnaround planning of a real airline. It shows two different workflow streams, one concerned with passengers and the cabin, the other with catering, fuelling and baggage handling.

As stated above, the information flow shown here represents a subset of the full turnaround flow. While it gives the full picture of the interactions and information exchanges, it has been stripped of some details for the

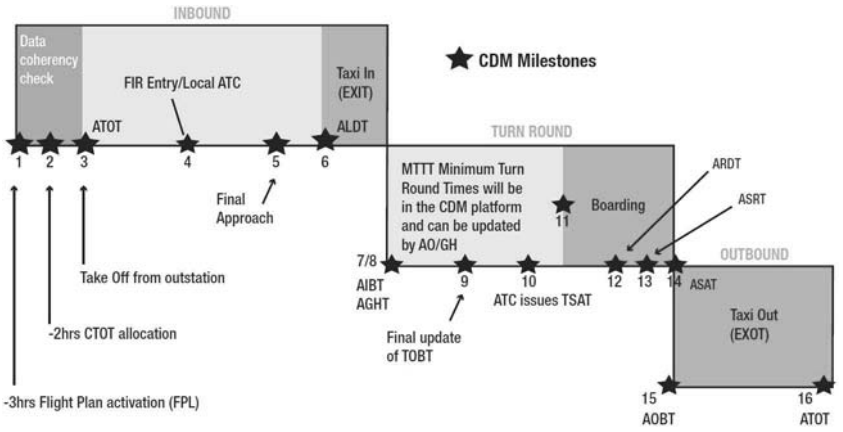


Fig. 6. Milestone approach.

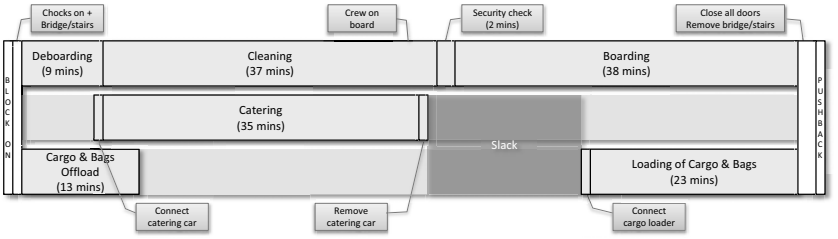


Fig. 7. Turnaround timeline example.

sake of readability and understandability of the test bed while still showing the interdependencies of the critical ICT infrastructure. The test bed represents the ICT systems of major players like EUROCONTROL's CFMU, an ATC provider, a ground handler and a CDM system (ACISP) and their interactions. Of the mentioned systems, ATC and the CFMU are to be considered critical from a safety as well as a business point of view. Disruption of the CFMU systems, e.g., by erroneous inputs from data providing systems, would reduce capacity in the European air traffic network with catastrophic effects for the aviation industry.

The test bed has a clear ICT focus while at the same time being coupled to physical processes and workflows at an airport. Many of the interactions taking place in the physical world are accompanied by message exchanges in the ICT system, for example, by time stamps being created in the ACISP. Scheduling and execution of essential services at an airport during aircraft turnaround are based on timing information stemming from players like the CFMU or ATC and being communicated via ICT systems. Hence there is a tight link and a strong similarity between the information and work flow in the ICT systems and the physical processes conducted.

The test case scenario and test bed have been developed in collaboration with Austro Control, the Austrian Air Navigation Service Provider (ANSP). CDM is a cornerstone of the business interests of Austro Control, which is actively involved in the implementation of airport CDM at a Vienna airport. As such, Austro Control can contribute realistic expertise to the test bed to ensure a close alignment of the scenario with the real world.

The test bed is a realistic example of a CDM system. This and the foundation of the EUROCONTROL CDM implementation manual make the results relevant for other airports in Europe. Note that the interactions between the ground handler and ramp services are not specified by EUROCONTROL, but the flow of information from the airport to the ATC stakeholders (which depends on the ground handler) is.

**9.5. Test bed information flow.** The previous section set the stage for the implementation of the test bed of SERSCIS. Embedding the actual test bed into the above described airport CDM scenario proves its real-world relevance. For the purpose of the test bed, an information flow between the CFMU, ATC, a ground handler, a fuelling company and the aircraft crew is chosen and shown in Fig. 8. The main ideas of the information flow are on the one hand a continuous update of ELDT, so that the fuelling service can prepare for its operation. On the other hand, based on completion time estimates from the fueller, the ground handler issues TOBTs. These form the basis for calculating TTOT and consequently being assigned a

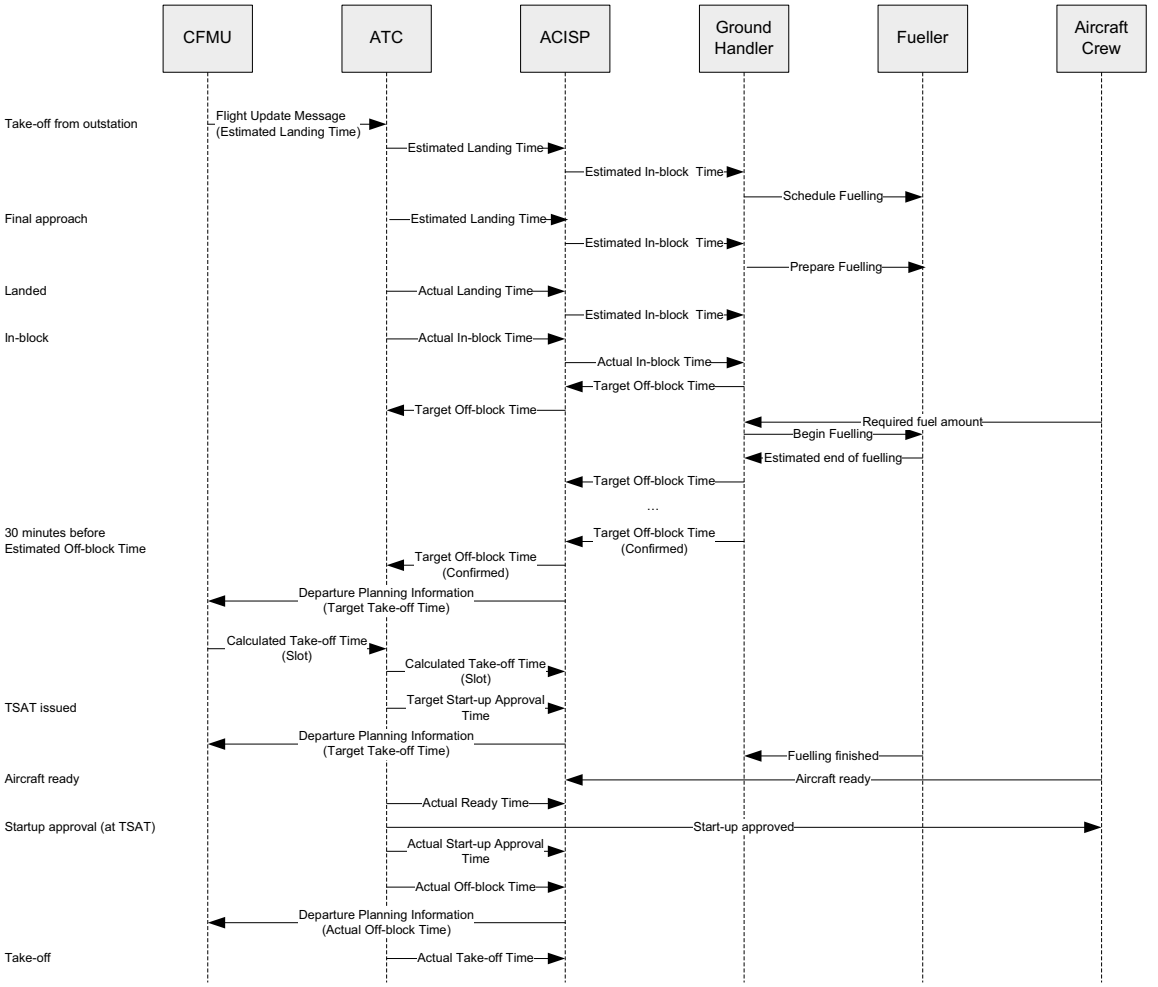


Fig. 8. Turnaround sequence diagram showing information flow.

'slot', or calculated take-off time (CTOT), by the CFMU.

The upper half of the sequence diagram in Fig. 8 deals with the inbound part of the turnaround. From the moment the aircraft takes off from the outstation, continuous updates of ELDT and, based on this, EIBT is derived from either the CFMU or local ATC systems. Apart from storing these values in the ACISP, they are used by the ground handler and the ramp services (the fueller in this example) to schedule and prepare the turnaround. When the aircraft enters the final approach and lands, the EIBT estimates are improved until the aircraft finally goes in-block.

Then the actual turnaround starts, which is shown in the lower half of the diagram. During this, two main interrelated flows take place. On the one hand, the ground handler communicates with the fueller to determine the estimated end of the turnaround (TOBT) and updates this value in the ACISP. This is based on estimates by the fueller (and other ramp services) of when they will finish.

On the other hand, there is the interaction between CFMU, ATC and the ACISP. As explained in the above sequence, the ground handler updates TOBT of the aircraft whenever a significant event in the turnaround changes the expected ready time of the aircraft. Using TOBT, the ACISP calculates the earliest possible take-off time of the aircraft, called TTOT.

Upon confirmation of TOBT (30 minutes before EOBT in the Vienna implementation) the ACISP sends the information to the CFMU and in turn receives a slot (CTOT) for the flight. ATC uses this information for sequence planning and issues a target start-up approval (TSAT) for the aircraft. Apart from the slot information, the sequencing takes into account a number of parameters including

- airline preferences,
- queue lengths at the holding points,
- taxiway blocking,
- adverse conditions such as construction on taxiways.

When the aircraft is ready, the pilot informs ATC. At pre-assigned TSAT for the aircraft, ATC issues a start-up approval. This information flow is depicted in Fig. 9.

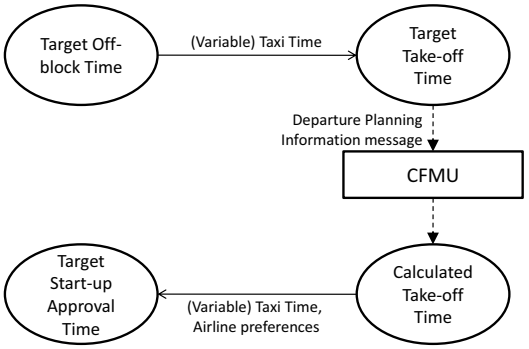


Fig. 9. CFMU-ATC communication and TSAT calculation.

The information flow thus incorporates the local airport dimension in the interaction between ATC, the ACISP as the information broker, the ground handler and the fueller and other ramp services. The European dimension is represented in this test bed as well by the interaction between the CFMU, ATC and the ACISP. Yet, these two dimensions are not isolated but closely interact. Similarly, the ICT systems of all actors in both dimensions are interconnected and exchange information.

Through this interconnection there is the risk of failure propagation ('cascading faults'). A disruption in one service at the airport will trigger changes at the local level (TOBT) that potentially induce changes at a European level (CTOT). Worse than the actual disruption of a service could be the effects of an attack on ICT systems of local services with the purpose of providing false values for finishing times. A coordinated attack to several services, probably at several airports, would cause ripples of disruption in the entire European air traffic network.

Finally, it is important to note here that the sequence diagram depicted in Fig. 8 shows the physical flow of information, which includes the ACISP as an intermediate and information broker. The corresponding logical flow takes place between real actors like ATC and ground handlers without an intermediate ACISP. This distinction between logical (end-to-end) and actual (transmission path) communications may be significant when considering possible reconfiguration to maintain resilience in the event of failures.

## 10. How SERSCIS helps

The approach used in SERSCIS is to exploit ICT systems modelling and the adaptive nature of service oriented architectures to contain problems by adapting the ICT network. This may be done by human operators using insights supported by SERSCIS technology, or in some cases through autonomic management of the ICT services through which information is exchanged.

We further assume that, because each actor has their own business agenda, none of them is in overall control of the information exchange network. Each actor manages their information flows (and associated physical processes) according to their own needs, but subject to agreements with other actors on how their information systems and physical processes will interact. These SLAs play a key role in the SERSCIS approach, because they provide the means to manage interdependencies and relate individual actor performance to that of the air traffic management network as a whole.

We must also bear in mind that no ICT management methodology can ever prevent disruption to airport operations and/or air traffic movements. Our goal is therefore to minimise the effects of such disruption, and ensure their consequences remain manageable. We have a special focus on preventing problems (including malicious attacks) at a local level from causing wider disruption through non-local (inter-actor) information interdependencies.

There are three basic mechanisms available by which SERSCIS can help to manage problems and make the system and its components more dependable:

- by identifying potential problems through system modelling, and redesigning the system (and the associated model) to eliminate the problem or prevent it from having any impact;
- by detecting a one-off problem during an aircraft turnaround, and managing its impact through autonomic workflow recomposition or a service management action;
- by detecting a recurrent or systematic problem and managing its impact on future aircraft turnarounds through system reconfiguration.

The first option is really only available to system designers and developers during system design (or redesign). The value of SERSCIS innovations here arises through the contribution to system modelling capabilities, e.g., the use of semantic domain or dependability models developed to model dynamic interdependent systems.

The second option can be used to mitigate problems when they arise during operation. For this to be effective, the corrective action must have an immediate effect and improve the dependability of information flows about the aircraft affected by the problem. This will almost certainly depend on the problem or at least its impact on information flows having been foreseen. If the problem arises in the ICT domain and can be detected at an early stage, it may be possible to mitigate its impact by applying ICT domain mitigation strategies, e.g., by using an alternative provider of data if the data are unexpectedly unavailable

when needed. However, if the problem arises in the physical domain and has only a secondary impact on the information flows, it is unlikely that SERSCIS will be able to mitigate the effects. For example, suppose there is a passenger no-show, but the baggage handling crew unexpectedly fails to return promptly to offload their bag. The aircraft will be delayed and may miss its allocated take-off slot. Since the problem was not foreseen, at least some of the information exchanged about this aircraft will have been invalid. But SERSCIS cannot detect this until it is too late to mitigate the effect (a missed slot) for this aircraft.

SERSCIS can be of value in detecting when a problem is not in fact 'one off', i.e., when it is a recurrent (but previously unforeseen) failure scenario, or a symptom of some deeper systemic problem. This is where the third type of mitigation strategy can help—systemic adaptation. For example, suppose that baggage handlers are regularly unable to respond promptly to deal with an unattended bag. SERSCIS could then alert the affected actor (the ground handler), allowing system changes to deal with the consequences. One solution might be to add a policy requiring baggage handlers to guarantee how long they will take to unload an unattended bag. Another option could be to reconfigure the workflow so that the ground handler does not provide a TOBT update (which results in the CFMU's issuing a take-off slot) until there are no unattended bags present on the aircraft. A third option might be to provide an update to TOBT early in the workflow, but specifying a later time to allow for a possible unattended baggage handling delay.

Such systemic mitigation strategies will not mitigate the impact of a problem on an individual aircraft turnaround, but should improve dependability over more extended periods of operation, e.g., one or more days. It is also important to recognise that systemic mitigation will usually involve human intervention by an operator representing the affected actor. Finally, note that autonomic 'in the loop' and systemic adaptations must be applied locally by an affected actor, in this case the ground handler. These actions will typically involve a trade off between the actor's own business goals (e.g., to minimise aircraft turn around time) and global dependability objectives (e.g., to minimise the number of missed slots).

**10.1. Turnaround scenario.** Figure 10 shows the services involved in the test case and how SERSCIS components are deployed. Each organisational/operational boundaries thus represent a service provider. SERSCIS system models, decision support, governance and orchestration components are installed at the ground handler. The SERSCIS framework can only directly access services and resources local to the ground handler. Access to external resources is via service requests. Other service providers (airline, fuel and catering companies) may or may not have SERSCIS. Figure 10 does not show all the SERSCIS features at all these providers where they may be used.

The fuel service in the test case is assumed to be one of many available in the service registry. The exact one that is used is chosen at runtime based on availability, price or any other QoS criteria specified in the SLA. The relationship with the catering company is managed by a long-term contract and hence is likely to be the only resource in the service registry from which the workflow composer can choose.

If the other service providers in Fig. 10 (airline, fuel company, catering company) are also SERSCIS enabled, their services will have their own governance and orchestration components, which provide automated management, and their own SERSCIS-assisted operator, who provides human and assisted management (the latter equipped by SERSCIS decision support tools). Without SERSCIS components, the SLA and resource management (including maintaining a repository of available resources) and workflow orchestration are a manual process, which may not even be consistent between organisations. With SERSCIS, not only are these processes computer assisted, but they can be made mutually consistent between service providers by sharing aspects of the overall system model and ontology.

**10.2. Service disruption scenarios.** The functioning of the services involved in the orchestration of the turnaround are typically subject to many kinds of threats that could cause disruption to the turnaround. In order to cover the SERSCIS mechanisms as comprehensively as possible, we distinguish three basic types of threats according to their recurrence and the countermeasure that SERSCIS supports:

**(M1)** One-off threats or failures, for which SERSCIS can help to mitigate the effects.

**(M2)** Recurring failures, for which SERSCIS can support the mitigation by systematic adaptation.

**(M3)** System problems identified in modelling and prevented from happening by redesigning the system.

From a phenomenal cause point of view, failures can be induced by physical (C1) or by ICT related compromises (C2). However, in both cases the primary concern is the impact on, and the usage of, the ICT facilities to mitigate the threats.

Three failure scenarios are devised to cover the evaluation goals listed above.

**10.2.1. Compromise of ramp service availability.** This scenario covers recurrence and countermeasures M1 and M2 as well as cause C1. Resources needed by the

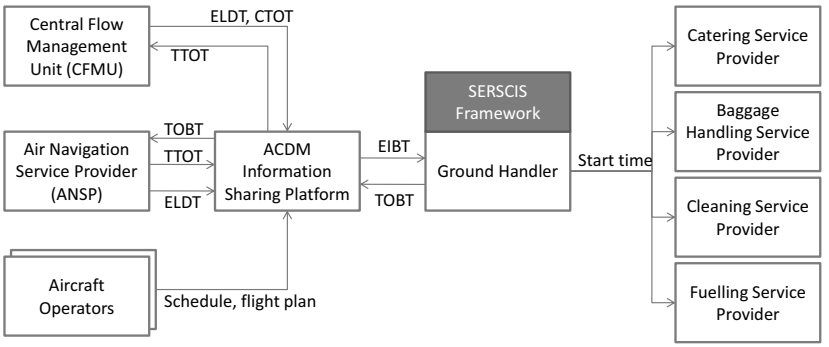


Fig. 10. SERSCIS components and services in the A-CDM test case.

ramp service may be compromised by an unexpected external (physical world) event. For example, cleaning crews may go on strike, fuel delivery trucks may break down, fuel supplies may be interrupted. This may cause ramp services to be delivered late or not at all.

Such failures will have an impact on the quality of information provided by the ground handler to the ACISP and thence to the CFMU. The ground handler calculates the estimate of when an aircraft will be ready to leave based on the availability of ramp services as communicated ahead of time by their service providers. If ramp services are then late or fail to appear (forcing the ground handler to reschedule the turnaround), the previous estimate of when the aircraft can leave will be wrong. In many cases this will force the aircraft to miss its take-off slot allocated by the CFMU. If too many aircraft miss their slots, this will adversely impact the capacity of European airspace (not just at the affected airport).

SERSCIS allows problems of this type to be detected by the ground handler as soon as a ramp service fails to appear. If the scheduling of ramp services is handled by coupling ICT systems via a service-oriented architecture, it should be easy to switch to an alternative service provider. SERSCIS can support this through the use of adaptive service orchestration technology, but this will not make the original estimate of aircraft take-off time more accurate. However, SERSCIS can provide a means to classify ramp service failure events according to their likelihood of recurrence, based on a model of their extended interdependencies. This allows the ground handler to adapt their processes for ramp service selection or for calculating the expected aircraft ready time for future aircraft allowing for the identified disruption.

**10.2.2. Physical distributed DoS attack.** A passenger who has checked in luggage does not show up for boarding. Consequently, his luggage needs to be unloaded. In its simple form this is a scenario handled by countermeasure M1 (alternative workflow applied). It is caused by type C1. A physical DoS attack compromises the availability of physical resources, which leads to the ICT requirement to detect and resist such an attack.

Realising the potential disruption that a passenger with a checked bag that 'no shows' can cause, an attacker organises a group of individuals to try to disrupt flight operations through a coordinated attack. A large number of people are convinced to knowingly check in bags and fail to show up at the gate when called.

All attackers in this scenario are passengers registered for different flights. A distributed attack across multiple flights is potentially disruptive on a wide scale, given that detection of an unattended bag must be made repeatedly at different gates and at different times. The delay to each flight owing to offloading the unattended bag is likely to create a disturbance that is only notable from a 'system-level' perspective (i.e., from the perspective of an airline across all of its flights, the airport or the ATC).

Viewed in isolation, the passenger no-shows appear no different to genuine no-shows and can be handled fairly easily. However, in combination they represent a significant threat to the ability of the airport to continue functioning, as many flights place greater demand on a shared resource (the baggage handlers), which may not be able to cope with the increased and unexpected load. Furthermore, the CFMU will have to revise many of the take-off slots, resulting in potential delays at destination airports.

SERSCIS should be able to detect the number of TOBT updates resulting from this attack, representing the anomalous rate of unattended bags, and alert the actors allowing them to change their procedures. For example, the GH might delay conformation of TOBT until the passenger check is complete—this will delay take-off, but prevents the problem from cascading into the ATM network.

**10.2.3. Degraded ACISP dependability.** The ACISP is critical to the A-CDM solution. Therefore, its performance and dependability is critical in terms of maintaining the integrity of the information provided to it, making

that information available to others in a timely fashion, and notifying CFMU, ATC and AO/GH of inconsistencies in the information. There are thus many ways in which the ACISP could fail, some of which are obvious, while others may not be immediately apparent to users of the system. If the ACISP exhibits degraded performance and passes on data after some delay, this could, for example, be caused by an ICT DoS attack that overloads the ACISP. Hence this is a failure caused by type C2. In its first occurrence this will be treated as a one-off event with countermeasures of type M1. A deeper evaluation could reveal malicious causes for this behaviour and trigger actions of type M2 at a larger scale.

In this scenario, we can consider the situation in which the failure mode of the ACISP involves flight information to be updated late. That is, information providers such as the CFMU, ATC and the ground handler put updated estimates of times relating to flight milestones (e.g., ELDT, TOBT). The ACISP calculates times derived from these estimates, such as EIBT and TTOT. Consider,for example, that the underperformance of the ACISP causes a long delay between the provision of TOBT by the GH and the availability of this updated value and the derived value for TTOT to other users. ATC and the CFMU will see the update eventually (as the service has not failed entirely), but the delay will restrict the amount of time they have to react to any problems (e.g., if the flight cannot make its assigned slot). Furthermore, the GH may be blamed for the late update of TOBT if the failure does not affect information stored for all flights.

The role of SERSCIS in this scenario is to detect the degraded ACISP performance by monitoring against dependability agreements. SERSCIS then addresses the deficit by allocating more ICT resources, or alerts the (human) operator if this is not possible. SERSCIS should also allow other operators to mitigate the effect of the degraded performance, e.g., by using alternative communication paths for time-critical information exchanges. It should also be possible for the ACISP or CFMU to compensate for the slow communication, e.g., by adapting internal workflows to allow more time between reported TTOT and issued CTOT, so any exceptions have time to propagate back to the CFMU.

## 11. Conclusion

A high-level service architecture has been introduced for a SERSCIS framework to support and manage critical infrastructure ICT services. This allows risks in an operating critical infrastructure to be managed by augmenting current 'slow' human-initiated management with automated and assisted management of ICT components and services. The service architecture maps to the lifecycle of an SLA: the definition of a service, the description of SLA templates, the negotiation of the SLA, and the selection and invocation of services as part of a workflow (including renegotiation/revocation of the SLA). To create and execute such a workflow comprising distributed services while maintaining an overall level of dependability requires models of information regarding performance characteristics, threats and countermeasures. The SERSCIS architecture takes a layered approach to help solve the conceptual integration, while the use of common components (a resource registry and ESB) addresses the pragmatics of integration and allows for loose coupling between the components.

## Acknowledgment

The research leading to these results has received funding from the European Community's 7th Framework Programme under the grant agreement no. 225336, SERSCIS.

## References

Aguado, V.M. (2009). *Airport CDM Guide*, EUROCONTROL, Brussels.

Balducelli, C., Di Pietro, A., Lavalle, L. and Vicoli, G. (2008). A middleware improved technology (MIT) to mitigate interdependencies between critical infrastructures, *in* R. de Lemos, F. Giandomenico, C. Gacek, H. Muccini and M. Vieira (Eds.), *Architecting Dependable Systems*, Lecture Notes in Computer Science, Vol. 5135, Springer, Berlin/Heidelberg, pp. 28–51.

Benkner, S. and Engelbrecht, G. (2006). A generic QoS infrastructure for grid web services, *Proceedings of the International Conference on Internet and Web Applications and Services/Advanced International Conference on Telecommunications, Guadeloupe, French Caribbean*, p. 141.

Boniface, M.J., Leonard, T.A., Surridge, M., Taylor, S.J., Finlay, L. and McCorry, D. (2005). Accessing patient records in virtual healthcare organisations, *Proceedings of eChallenges, Ljubljana, Slovenia*.

Boniface, M.J., Phillips, S.C. and Surridge, M. (2006). Grid-based business partnerships using service level agreements, *Proceedings of the Cracow Grid Workshop, Cracow, Poland*, pp. 165–175.

Boniface, M., Phillips, S., Sanchez-Macian, A. and Surridge, M. (2009). Dynamic service provisioning using GRIA SLAs, *in* E. di Nitto and M. Ripeanu (Eds.), *Service-Oriented Computing—ICSOC 2007 International Workshops, Revised Selected Papers*, Lecture Notes in Computer Science, Vol. 4907, Springer, Berlin/Heidelberg, pp. 56–67.

Froihofer, L., Goeschka, K.M. and Osrael, J. (2007). Middleware support for adaptive dependability, *in* R. Cerqueira and R.H. Campbell (Eds.), *Proceedings of the 8th International Middleware Conference*, Lecture Notes in Computer Science, Vol. 4834, Springer, Berlin/Heidelberg/New York, NY, pp. 308–327.

*GRIA.org* (n.d.). `http://www.gria.org`.

Hasselmeyer, P., Koller, B., Schubert, L. and Wieder, P. (2006). Towards SLA-supported resource management, *in* M. Gerndt and D. Kranzlmüller (Eds.), *Proceedings of the 2nd International High Performance Computing and Communications Conference*, Lecture Notes in Computer Science, Vol. 4208, Springer, Berlin/Heidelberg, pp. 743–752.

Hasselmeyer, P., Mersch, H., Koller, B., Quyen, H.N., Schubert, L. and Wieder, P. (2007). Implementing an SLA negotiation framework, *Proceedings of the eChallenges Conference (e-2007), The Hague, The Netherlands*, pp. 24–26.

Hovestadt, M. (2005). Fault tolerance mechanisms for SLA-aware resource management, *Proceedings of the 11th International Conference on Parallel and Distributed Systems, Fukuoka, Japan*, Vol. 2, pp. 458–462.

Hovestadt, M. (2006). *Service Level Agreement Aware Resource Management*, Ph.D. thesis, University of Paderborn, Paderborn.

Kramer, J. and Magee, J. (2009). A rigorous architectural approach to adaptive software engineering, *Journal of Computer Science and Technology* **24**(2): 183–188.

McKee, P., Taylor, S.J., Surridge, M., Lowe, R. and Ragusa, C. (2007). Strategies for the service market place, *in* J. Altmann and D.J. Veit (Eds.), *Proceedings of GECON 2007—Grid Economics and Business Models*, Lecture Notes in Computer Science, Vol. 4685, Berlin/Heidelberg, pp. 58–70.

Michlmayr, A., Rosenberg, F., Leitner, P. and Dustdar, S. (2008). Advanced event processing and notifications in service runtime environments, *Proceedings of the 2nd International Conference on Distributed Event-based Systems, Rome, Italy*, pp. 115–125.

Mikulik, J. and Zajdel, M.A. (2009). Automatic risk control based on FSA methodology adaptation for safety assessment in intelligent buildings, *International Journal of Applied Mathematics and Computer Science* **19**(2): 317–326, DOI:102478/v10006-009-0027-1.

Naqvi, S., Mouton, S., Massonet, P., Silaghi, G., Battre, D., Hovestadt, M. and Djemame, K. (2008). Using SLA based approach to handle sabotage tolerance in the grid, *in* T. Priol and M. Vanneschi (Eds.), *From Grids to Service and Pervasive Computing*, Springer, Berlin/Heidelberg, pp. 153–162.

*NextGRID* (n.d.). `http://www.nextgrid.org`.

Radetzki, U., Boniface, M. and Surridge, M. (2007). Contextualized B2B registries, *in* E. di Nitto and M. Ripeanu (Eds.), *Proceedings of Service-Oriented Computing (ICSOC 2007)*, Lecture Notes in Computer Science, Vol. 4749, Springer, Berlin/Heidelberg, p. 506.

Sommerville, I., Storer, T. and Lock, R. (2007). Responsibility modelling for contingency planning, *Proceedings of the Workshop on Understanding Why Systems Fail, Edinburgh, UK*.

Surridge, M., Taylor, S., De Roure, D. and Zaluska, E. (2005). Experiences with GRIA—Industrial applications on a web services grid, *First International Conference on e-Science and Grid Computing, Melbourne, Australia*, pp. 98–105.

Verissimo, P., Deswarte, Y., Bondavalli, A., Neves, N.F., El Kalam, A.A., Daidone, A. and Correia, M. (2008). The CRUTIAL architecture for critical information infrastructures, *in* R. de Lemos, F. Giandomenico, C. Gacek, H. Muccini and M. Vieira (Eds.), *Architecting Dependable Systems*, Lecture Notes in Computer Science, Vol. 5135, Springer, Berlin/Heidelberg, pp. 1–27.

**Martin Hall-May** holds an M.Eng. (University of Bristol) and a Ph.D. (University of York) in computer science. He is currently a senior research engineer at the University of Southampton IT Innovation Centre, where he leads research and integration activities in the EC FP7 SERSCIS project. His current research interests include the dependability of service oriented architectures and automated system management. Prior to joining the centre, Martin carried out research into large-scale system safety and the implications of deploying autonomous vehicles in safety-critical applications.

**Mike Surridge** is the research director of the IT Innovation Centre. He has a Ph.D. in theoretical physics, and over 20 years experience in IT research mainly on distributed computing. His current focus is on dynamic service-oriented architectures, especially dynamic federated security and adaptation. He coordinated the EC FP5 GRIA project, which produced the first business-oriented grid middleware supporting business-friendly security federation and resource management models. He is currently the coordinator of the EC FP7 SERSCIS project on dynamic SOAs for critical infrastructure protection, and provides advice to research projects and industry on dynamic SOAs, business models and security. He was a co-founder of the UK e-Science Security Task Force, and is currently a co-chair of the NESSI Working Group on Trust, Security and Dependability.

**Roman Nossal-Tüyeni** holds an M.Sc. and a Ph.D. degree in computer science (embedded systems) from the Vienna University of Technology, an M.B.A. from Danube University Krems, and an M.A. in business from the University of Applied Science in Eisenstadt, Austria. After spending several years as an assistant professor at the Vienna University of Technology, he joined Siemens Automotive in Regensburg, Germany, where he participated in the development of the OSEKtime operating system. Thereafter he worked as a project manager for Bosch and then as a product manager for DECOMSYS (now Elektrobit Austria), participating in the development of the FlexRay communication system. Currently, he is a service development manager at the technical department for Austro Control, the Austrian Air Navigation Service Provider. He is also involved as a project manager in the European Commission's SESAR programme aiming at developing Europe's next generation of air traffic management.

Received: 28 June 2010
Revised: 21 November 2010
Re-revised: 8 January 2011